FASTER, CHEAPER, BETTER: MULTI-OBJECTIVE HYPERPARAMETER OPTIMIZATION FOR LLM AND RAG SYSTEMS

Matthew Barker
Trustwise AI
matthew@trustwise.ai

Andrew Bell
New York University
alb9742@nyu.edu

Evan Thomas
Trustwise AI
evan@trustwise.ai

James Carr
Trustwise AI
jamie@trustwise.ai

Thomas Andrews
Trustwise AI
thomas@trustwise.ai

Umang Bhatt
New York University
The Alan Turing Institute
umangbhatt@nyu.edu

ABSTRACT

While Retrieval Augmented Generation (RAG) has emerged as a popular technique for improving Large Language Model (LLM) systems, it introduces a large number of choices, parameters and hyperparameters that must be made or tuned. This includes the LLM, embedding, and ranker models themselves, as well as hyperparameters governing individual RAG components. Yet, collectively optimizing the entire configuration in a RAG or LLM system remains under-explored—especially in multi-objective settings—due to intractably large solution spaces, noisy objective evaluations, and the high cost of evaluations.

In this work, we introduce the first approach for multi-objective parameter optimization of cost, latency, safety and alignment over entire LLM and RAG systems. We find that Bayesian optimization methods significantly outperform baseline approaches, obtaining a superior Pareto front on two new RAG benchmark tasks. We conclude our work with important considerations for practitioners who are designing multi-objective RAG systems, highlighting nuances such as how optimal configurations may not generalize across tasks and objectives.

1 INTRODUCTION

Retrieval Augmented Generation (RAG) has emerged as a popular technique for improving the performance of Large Language Models (LLMs) on question-answering tasks over specific datasets. A benefit of using RAG pipelines is that they can often achieve high performance on specific tasks *without* the need for extensive alignment and fine-tuning (Gupta et al., 2024), a costly and time-consuming process. However, the end-to-end pipeline of a RAG system is dependent on many parameters that span different components (or modules) of the system, such as the choice of LLM, the embedding model used in retrieval, the number of chunks retrieved and hyperparameters governing a reranking model. Examples of choices, parameters, and hyperparameters that are often made or tuned when implementing a RAG pipeline are listed in Table 1. Importantly, the performance of a RAG pipeline is dependent on these choices (Fu et al., 2024), many of which can be difficult to tune manually. While those building RAG pipelines might avoid fine-tuning costs, they often spend time and resources on hyperparameter optimization (HO).

Despite this, there is little research exploring methods for collectively optimizing all the hyperparameters in a given LLM and RAG pipeline (Fu et al., 2024). Further, to the best of our knowledge, there is no work that addresses this challenge in *multi-objective settings*, where the RAG pipeline must achieve high performance across a range of objectives, like minimizing a system's inference time while maximizing its helpfulness. In this work, we aim to fill this gap by introducing an approach for collectively optimizing the hyperparameters of a RAG system in a multi-objective setting.

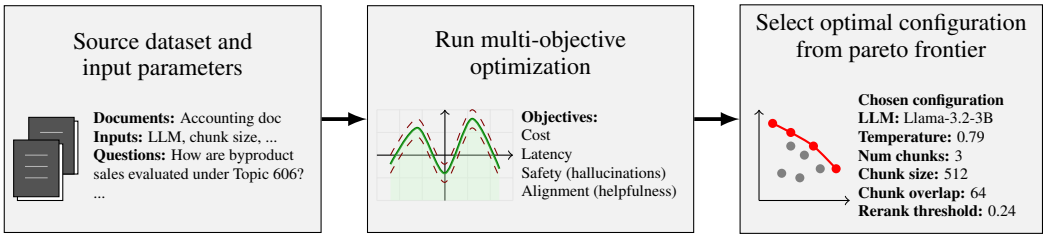

Figure 1: A high-level overview of our approach. First, we source the datasets that we will use to optimize our RAG pipeline, define the choices, parameters and hyperparameters that will be optimized over (see Table 1), and select the objectives for optimization (*e.g.,* cost, latency, safety, and alignment). Second, we introduce a train-test paradigm for evaluating the performance of RAG pipelines, and use Bayesian optimization (BO) to find the optimal parameter configurations. We find that using BO with the `qLogNEHVI` (Daulton et al., 2021; Ament et al., 2023) acquisition function is well-suited for this problem, since it is adapted for noisy objective evaluations and makes use of a single composite objective called *hypervolume improvement* that allows for an arbitrary number of objectives. Third, we explore the Pareto frontier of parameter configurations, finding the best solutions over different objectives.

The authors of this paper are from a mixture of both academia and industry, and this work was motivated by real-world challenges faced by industry practitioners. Use of RAG pipelines within industry often requires balancing multiple requirements which are in competition with one another. For example, at one financial services firm developing an in-house Q&A chatbot to support internal workflows, practitioners aimed to both maximize accuracy and minimize the time taken to generate a response. However, there is a tension in these two objectives: a RAG system utilizing larger models may yield more accurate responses, but consequently requires longer computation time. As another example, a large bank developing an external insurance policy Q&A chatbot was primarily concerned with the alignment of generated responses to policies and regulations. Objectives that we have observed practitioners frequently consider when building RAG pipelines include: cost, response latency, safety (hallucination risk), and alignment (response helpfulness).

Multi-objective HO over a RAG pipeline is particularly challenging for several reasons. First, RAG pipelines naturally have a high number of parameters, leading to a large solution space. We identify at least 15 example choices, parameters, and hyperparameters in Table 1. Even if one has just a handful of possible values for each choice, the parameter space becomes intractably large for simple algorithms like grid search (Bergstra & Bengio, 2012). Second, evaluating a RAG pipeline during the HO process is costly, with respect to both compute resources and time: it requires running the RAG system over multiple queries (where each iteration is bounded by the per-token inference time of the LLM), and then evaluating each output. Third, the objective evaluations can be noisy. The true characteristic of a RAG system cannot be computed directly, and requires sampling the evaluations from many queries. Relatedly, since in most cases LLMs are non-deterministic, combinations of hyperparameters need to be tested over multiple seeds.

Contributions. In this work, we make four main contributions:

(1) We introduce an approach for multi-objective optimization over a unique set of hyperparameters of a RAG pipeline, including choices for the LLM and embedding models themselves. Our approach implements a single composite objective value called the hypervolume indicator (Guerreiro et al., 2021), and uses Bayesian optimization with an acquisition function that allows for an arbitrary number of noisy objective functions (`qLogNEHVI`) (Daulton et al., 2021; Ament et al., 2023) to find the best RAG pipeline configuration.

(2) We empirically show the effectiveness of our approach to identify optimal RAG pipeline configurations across two tasks (one related to financial services Q&A, and another related to medical Q&A) using a train-test paradigm, as compared to random parameter choices and other baseline optimization approaches.

(3) We publicly release two novel benchmarks for evaluating RAG systems called `FinancialQA` and `MedicalQA`.[1] Importantly, these benchmarks more closely mimic industry RAG use-cases than currently available benchmarks, since the context must be retrieved at runtime from an available document, rather than being given in the dataset.

(4) We frame our discussion (Section 7) as guidance to practitioners who seek to improve their own RAG systems. We highlight two important considerations: the first is what we call "task dependence", meaning that an optimal configuration for a RAG pipeline on a task in a specific setting may not generalize to another setting. The second is "objective dependence", where objective evaluations follow different trends (or have no trend) across different configurations. Task and objective dependence can also compound, highlighting the challenge of collectively optimizing the parameters of a RAG system.

2 RELATED WORK

There has been a mixture of work separately addressing multi-objective optimization and hyperparameter optimization in LLM and RAG systems, which we summarize here. We provide further comparison with fine-tuning and model-merging approaches in Appendix A.

Hyperparameter optimization (HO). As many pieces of LLM and RAG pipelines have hyperparameters that must be tuned before deployment, there is a large body of work testing the efficacy of using HO to tune these systems.

Wang et al. (2023) propose a cost-based pruning strategy to hyperparameter tune LLM systems under budget constraints. They focus on hyperparameters like the type of model (*e.g.,* text-davinci-003, gpt-3.5-turbo, or gpt-4), the maximum number of tokens that can be generated in a response, the model temperature[2], and the model top-p.[3] They use a search method called BlendSearch (Wang et al., 2021), which combines Bayesian optimization and local search (Wu et al., 2021), to find the optimal combinations of parameters, and measure performance on the tasks APPS (Hendrycks et al., 2021), XSum (Narayan et al., 2018), MATH (Hendrycks et al., 2021), and HumanEval Chen et al. (2021). In comparison with our work, Wang et al. (2023) do not consider a RAG system.

Most related to our work, Kim et al. (2024) proposed AutoRAG, an open-source framework designed for RAG experimentation and hyperparameter optimization. They use a a greedy algorithm for selecting the hyperparameters governing RAG modules like query expansion, retrieval, passage augmentation, passage re-ranking, prompt making, and generating (the LLM). In concurrent work, Fu et al. (2024) proposed AutoRAG-HP, which frames hyperparameter selection as an online multi-armed bandit (MAB) problem. To carry out HO, they introduce a novel two-level Hierarchical MAB (Hier-MAB) method, where a high-level MAB guides the optimization of modules, and several low-level MABs search for optimal settings within each module. Significantly, our work is distinct from both Kim et al. (2024) and Fu et al. (2024) in that they do not consider multi-objective settings.

Multi-objective alignment. Several researchers have proposed methods for incorporating multiple objectives directly into the LLM fine-tuning and alignment processes. Li et al. (2020) developed an approach for multi-objective alignment from human feedback using scalar linearization. Mukherjee et al. (2024) expanded on that approach by developing an algorithm that finds a diverse set of Pareto-optimal solutions that maximize the hypervolume, given a set of objectives. Zhou et al. (2024) proposed a reward-function free extension called Multi-Objective Direct Preference Optimization (MODPO). The latter showed that MODPO can effectively find a Pareto-optimal frontier of fine-tuned models, trading off objectives like "helpfulness" and "harmlessness". While these works have demonstrated success in multi-objective LLM alignment, we focus on RAG pipelines and avoid aligning and fine-tuning models altogether.

[1] `https://huggingface.co/datasets/Trustwise/optimization-benchmark-dataset`

[2] A parameter for which low or high values sharpen or soften the probability distribution of a token being outputted by an LLM, respectively.

[3] A parameter that restricts the domain of tokens that can be outputted by an LLM to those whose cumulative probability is greater than p.

Table 1: Common choices, parameters, and hyperparameters that are often made/tuned when implementing RAG pipelines. **Bold** indicates a parameter that was optimized over in our experiments.

Domain	Parameters	Notes
System-level	**LLM model** **Embedding model**	*e.g.,* gpt-4, llama-3.1-8b, llama-3.1-70b For RAG pipelines, *e.g.,* text-embedding-ada-002
LLM controls	System prompt **Temperature** Top-k, top-p Max length of output	
Fine-tuning	Preference-tuning approach	*e.g.,* RLHF, DPO, KTO; these methods may also introduce hyperparameters that must be tuned, *e.g.,* DPO uses β, KTO uses $\beta, \lambda_U, \lambda_D$
	Parameter-Efficient Fine-Tuning (PEFT)	*e.g.,* LoRA, adapter modules (Houlsby et al., 2019); these methods may also introduce hyperparameters, *e.g.,* LoRA uses rank and scaling α
	Dropout rate Learning rate Training epochs	
RAG controls	Modules	Approaches for query expansion, retrieval, passage augmentation and re-ranking, and prompt making
	Chunk size **Number of chunks** **Chunk overlap** **Re-rank threshold** Retrieval size (top-k)	

3 Preliminaries and problem formulation

Multi-Objective Optimization (MOO). The goal of MOO is to find a solution $\mathbf{x} \in \mathcal{X}$ that maximizes (or minimizes) a set of objective functions, where $\mathbf{x} = [x_1, x_2, \dots x_l]$ corresponds to a series of input values, and $\mathcal{X}$ is said to be the solution space. We then define each objective as $f : \mathcal{X} \to \mathbb{R}$ and use $\mathbf{f} : \mathcal{X} \to \mathbb{R}^k$ to represent k objective functions. Using these definitions, we aim to solve the following optimization problem:

$$\max_{\mathbf{x} \in \mathcal{X}} \mathbf{f}(\mathbf{x}) := \max_{\mathbf{x} \in \mathcal{X}} [f_1(\mathbf{x}), f_2(\mathbf{x}), \dots f_k(\mathbf{x})] \tag{1}$$

Rather than identifying a single solution, MOO algorithms identify a set of non-dominated solutions (Deb et al., 2016). We use $\mathbf{f}(\mathbf{x}^*) \succ \mathbf{f}(\mathbf{x})$ to signify that $\mathbf{f}(\mathbf{x}^*)$ dominates $\mathbf{f}(\mathbf{x})$:

$$\forall i \in \{1, \dots, k\}, \quad f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*), \quad \text{and} \quad \exists j \text{ s.t. } f_j(\mathbf{x}) < f_j(\mathbf{x}^*) \tag{2}$$

Hence, and following Daulton et al. (2021), we define the *Pareto set* as $\mathcal{P}^* = \{\mathbf{f}(\mathbf{x}^*) \mid \mathbf{x}^* \in \mathcal{X}, \nexists\ \mathbf{x} \in \mathcal{X} \text{ s.t. } \mathbf{f}(\mathbf{x}) \succ \mathbf{f}(\mathbf{x}^*)\}$, and the corresponding *Pareto optimal solutions* as $\mathcal{X}^* = \{\mathbf{x}^* \mid \mathbf{f}(\mathbf{x}^*) \in \mathcal{P}^*\}$. In practice, the Pareto optimal set often consists of an infinite set of points. Given a set of observed solutions from $\mathcal{X}$, we aim to identify an approximate Pareto optimal set, $\hat{\mathcal{P}} \subset \mathbb{R}^k$, and its associated Pareto optimal solutions, $\hat{\mathcal{X}}$. We then use the hypervolume (HV) indicator, $\mathcal{HV}$, to evaluate the quality of $\hat{\mathcal{P}}$ given a reference point $\mathbf{r} \in \mathbb{R}^k$. Our optimization problem then becomes:

$$\arg\max_{\hat{\mathcal{P}}} \mathcal{HV}(\hat{\mathcal{P}} | \mathbf{r}) \tag{3}$$

In this work, we seek to find the Pareto-optimal combinations of parameters of a RAG system (those indicated in Table 1). We represent a *configuration* of a RAG system as a solution vector $\mathbf{x} \in \mathcal{X}$, where each value corresponds to a system parameter. We aim to find the solution set containing different RAG pipeline configurations that maximizes $\mathcal{HV}$ for the objectives in Section 4.2.

Bayesian Optimization (BO). Objective function evaluations herein are obtained using an entire RAG pipeline, meaning there is no single analytic expression or gradient available for solving Equation (3). BO is a derivative-free optimization method that works by constructing probabilistic surrogate models to capture the uncertainty of objective functions. The core methodology employs Gaussian process regression to model the unknown objective landscape, where each function f_i is represented as a stochastic process with a posterior distribution $p(f_i \mid \mathcal{D})$ conditioned on observed data (Williams & Rasmussen, 2006). Importantly, BO makes use of an acquisition function to balance exploration of unknown regions and exploitation of promising solutions. BO has been known to perform well compared to other optimization methods when objective functions are expensive to evaluate, as in our setting (Gramacy, 2020; Diessner et al., 2022; Guerreiro et al., 2021).

4 METHODOLOGY

Our approach works by defining a solution space (over the configurations of a RAG pipeline), objectives, and a train-test paradigm, then using BO to find the optimal configuration. BO is well-suited to exploit patterns in objective evaluations, for example the tendency for latency to increase with chunk size. We allow that the solution space has a mixture of continuous variables (*e.g.,* temperature of LLM) and categorical variables (*e.g.,* choice of LLM). In addition, we allow for constraints on the inputs, such as asserting that the chunk overlap must be less than the chunk size.

We make use of two state-of-the-art algorithms that implement and extend BO. The first, `qLogEHVI`, takes advantage of recent advances in programming models and hardware acceleration to parallelize multi-objective BO using the LogEI variant (Ament et al., 2023) of *expected hypervolume improvement* to guide the acquisition of new candidate solutions (Daulton et al., 2020). The second is `qLogNEHVI`, which extends `qLogNEHVI` by using a novel acquisition function that was theoretically motivated and empirically demonstrated to outperform benchmark methods in settings with noisy objective evaluations (Daulton et al., 2021). The noisy variant is particularly useful since the probabilistic nature of LLMs can cause noisy objective function evaluations. BO is also well-suited to exploit patterns in objective evaluations, for example the tendency for latency to increase with chunk size. Following Daulton et al. (2021), we initialize both BO algorithms with N_{init} points from a scrambled Sobol sequence.

We outline our proposed methodology in Algorithm 1, and provide implementation details in Section 5. We also provide a method of approximating each objective function in Appendix B.

Algorithm 1 Train-test multi-objective optimization of RAG or LLM system

Require: set of documents $\mathcal{D}$, solution space $\mathcal{X}$, reference point $\mathbf{r}$, number of iterations N, number of iterations for BO initialization N_{init}

$Q, Q_{\text{test}} \leftarrow$ `Generate`$(\mathcal{D})$ ▷ Generate train-test queries from documents
$H, H_{\text{test}} \leftarrow [\,], [\,]$ ▷ Start with empty train and test history
for $n = 1 ... N$ **do** ▷ Parameter optimization
  if $n \leq N_{\text{init}}$ **then**
    $\mathbf{x} \leftarrow$ `Sobol`$(H, \mathcal{X})$ ▷ Sobol sampling for N_{init} iterations
  else
    $\mathbf{x} \leftarrow$ `qLogNEHVI`$(H, \mathcal{X})$ ▷ `qLogNEHVI` acquisition function
  end if
  $\mathbf{f} \leftarrow$ `ObjectiveEvaluations`$(\mathbf{x}, Q)$ ▷ Evaluate solution on Q
  H.`append`$(\{\mathbf{x}, \mathbf{f}\})$ ▷ Save to history
  $\hat{\mathcal{X}}, \hat{\mathcal{P}} \leftarrow$ `ParetoOptimalSet`(H) ▷ Find Pareto-optimal set
  $\text{HV} \leftarrow \mathcal{HV}(\hat{\mathcal{P}}|\mathbf{r})$ ▷ Find hypervolume w.r.t reference point
  $\mathbf{f}_{\text{test}} \leftarrow$ `ObjectiveEvaluations`$(\mathbf{x}, Q_{\text{test}})$ ▷ Evaluate solution on Q_{test}
  H_{test}.`append`$(\{\mathbf{x}, \mathbf{f}_{\text{test}}\})$ ▷ Save to history
  $\hat{\mathcal{X}}_{\text{test}}, \hat{\mathcal{P}}_{\text{test}} \leftarrow$ `ParetoOptimalSet`(H_{test}) ▷ Find test Pareto-optimal set
  $\text{HV}_{\text{test}} \leftarrow \mathcal{HV}(\hat{\mathcal{P}}_{\text{test}}|\mathbf{r})$ ▷ Find test hypervolume w.r.t reference point
end for
$X_{\text{opt}} \leftarrow$ `SelectOptimalConfig`$(\hat{\mathcal{X}}_{\text{test}}, \hat{\mathcal{P}}_{\text{test}})$ ▷ Select optimal configuration(s)

4.1 GENERATING TRAIN-TEST QUERIES

A frequent challenge encountered by the authors in industry is a lack of existing queries for RAG Q&A tasks. To this end, we use LLMs to help generate queries from the data, which may be PDF documents or large documents of text. LLMs are a reasonable choice for this approach because they have been shown to be effective at generating synthetic data (Long et al., 2024). For our `FinancialQA` dataset described in Section 5.1, we generate train and test queries using an LLM (GPT-4o). We publicly release the synthetic questions as part of our datasets, and provide the prompt used to generate the questions in Appendix D.

4.2 OBJECTIVES

Motivated by experiences in industry, we consider four objectives that practitioners commonly consider important: safety, alignment, cost and latency.

Safety. In this work, we use the term "safety" to refer to *hallucination risks*, or the risk that a RAG pipeline will return false information to the user. Hallucinations can cause significant downstream harm, particularly in high-stakes domains such as healthcare. In our experiments, we evaluate safety using the *faithfulness* metric defined in the Trustwise API.[4] Like previous work (Min et al., 2023; Es et al., 2023), faithfulness detects hallucinations by evaluating whether or not the response from a RAG system is supported by the context. The response is split into individual, "atomic" claims that are verified with respect to the context. Scores of these verifications are then aggregated into a single faithfulness score between 0 and 100 for each response, where 100 represents a completely "safe" response with no hallucinations.[5]

Alignment. While hallucination risks are an immediate concern, the alignment of a response is often just as important in enterprise use-cases. To evaluate alignment, we follow the definition of *helpfulness* popularized by Anthropic (Bai et al., 2022). We measure alignment using the *helpfulness* metric as implemented in the Trustwise API which judges how useful, detailed and unambiguous a response is. This metric assigns a score between 0 and 100 for each response, where a higher score indicates a more helpful response.

Cost. To calculate the cost of an evaluation, we consider all the components of a RAG pipeline, including the query embedding cost, reranker embedding cost, LLM input token cost and LLM output token cost. Importantly, the cost of a RAG pipeline is a function of its configuration:

$$\begin{aligned}\text{cost} = {} & \text{number of query tokens} \times \text{cost per embedding token} \\ & + \text{number of context tokens} \times \text{cost per reranker token} \\ & + \text{number of prompt input tokens} \times \text{cost per LLM input token} \\ & + \text{number of output tokens} \times \text{cost per LLM output token}\end{aligned} \tag{4}$$

The embedding cost per query token, reranker token cost, LLM input token cost, and LLM output token cost are based on the specific choices of those models, as well as the hardware being used to run the RAG pipeline. In enterprise use-cases, these costs may also include overhead and maintenance.

Latency. We define latency as the time it takes for a complete end-to-end run of the RAG pipeline, from the moment an initial query is sent to the system to the moment a full response is returned to the user. As with cost, we can calculate the latency of a system as a function of its configuration:

$$\text{latency} = \text{embedding latency} + \text{reranker latency} + \text{LLM response latency} + \text{evaluation latency} \tag{5}$$

We note that response evaluations can take as long as, or longer than, response generation and thus the end-to-latency is vital to consider in enterprise settings where evaluations are a requirement.

[4] Trustwise documentation available at: `https://trustwise.ai/docs`

[5] For those aiming to replicate our approach, there are open-source alternatives for evaluating safety that could be used instead, such as LlamaGuard (Inan et al., 2023).

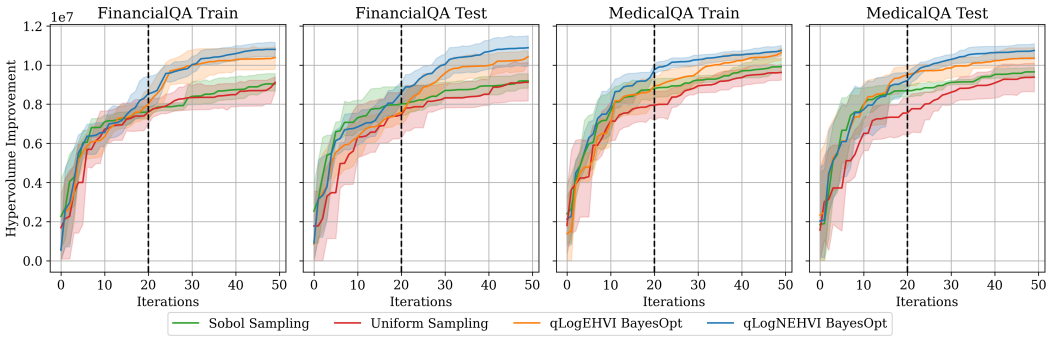

Figure 2: HV improvement on train and test splits for both datasets. Our proposed acquisition function for BO (`qLogNEHVI`) outperforms its noiseless variant (`qLogEHVI`) and both BO algorithms perform significantly better than the baselines. There is a noticeable increase in HV after iteration 20 (dotted line), indicating the end of Sobol sampling initializations for the BO algorithms, and the start of acquisition function-guided selections.

5 Experiments

We tested our optimization approaches on two RAG tasks from different industries. We use a standard RAG setup with retrieval using vector embeddings, a reranker to filter out unnecessary context chunks, and an LLM prompted to generate a response using the context. The exact configuration solution space we use is given in Appendix C.

We optimize for the four objectives of cost, latency, safety and alignment. We run BO using the train question set, and then report results on a held out test set. We use *Ax* (Bakshy et al., 2018) and *BoTorch* (Balandat et al., 2020) to run and manage experiments. For all algorithms we use 50 total iterations, and for BO methods, the first 20 iterations are chosen using Sobol sampling. In MOO, a reference point is used to calculate the HV improvement, and represents the minimum acceptable solution. Based on our industry experience, we use a reference point with cost of $2000 per million queries, latency of 20s per query, safety of 50 and alignment of 50. This choice of reference point prevents degenerate solutions (*e.g.,* a degenerate RAG system which does not retrieve any chunks) from contributing to the HV improvement.

Throughout this work, we report the cost in USD per million queries ($ / million queries) and latency in seconds (s). Safety and alignment scores are dimensionless and range from 0 to 100. Cost and latency are objectives to minimize, while safety and alignment are objectives to be maximized.

5.1 Datasets

Existing datasets for Q&A tasks generally exist in the form of (question, context, answer) triplets. However, in industry RAG use-cases, the context is often retrieved at runtime from a set of documents. As such, we want to include the retrieval of the context as part of of our evaluation. To this end, we adapt two known tasks to fit the needs of our experimental setup[6]:

FinancialQA. This task uses a publicly available document covering revenue recognition from a leading global accounting firm.[7] The document includes more than 1000 pages of text, representing a significant Q&A context retrieval challenge. Since the document does not come with a set of questions, we synthetically generate 50 questions by prompting GPT-4o, using the method described in Section 4.1. We then randomly split this set into 30 train and 20 test questions.

[6]We publicly release the train and test questions for these documents at `https://huggingface.co/datasets/Trustwise/optimization-benchmark-dataset`

[7]Financial document available at: `https://kpmg.com/kpmg-us/content/dam/kpmg/frv/pdf/2024/handbook-revenue-recognition-1224.pdf`

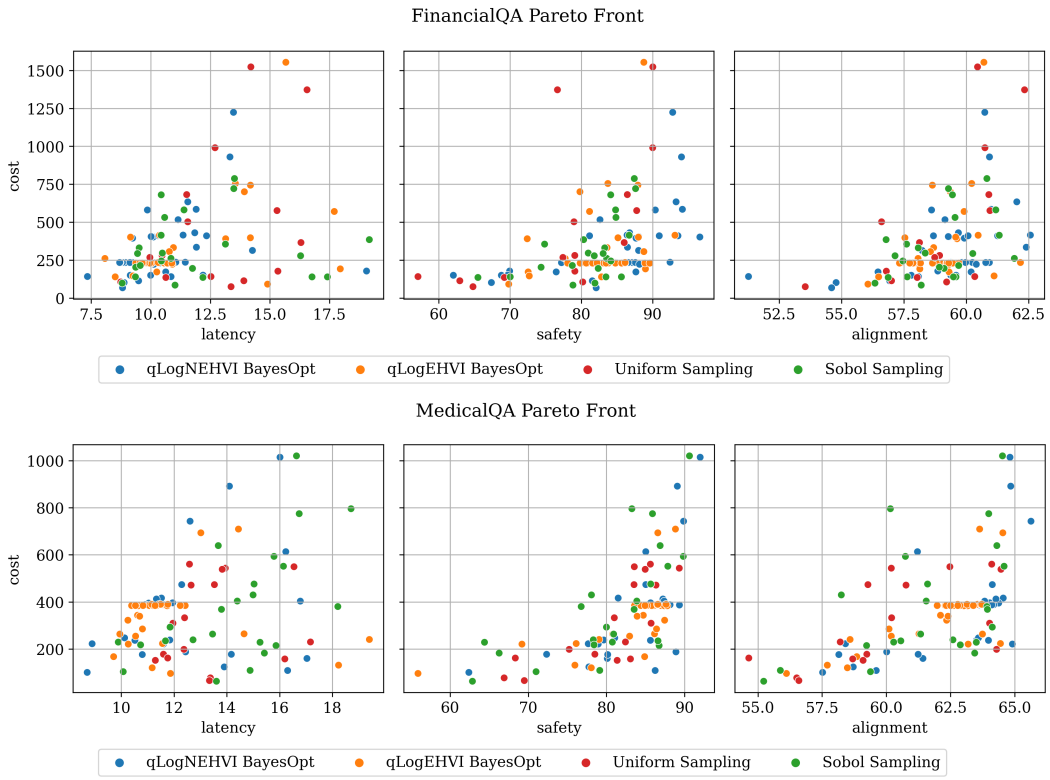

Figure 3: 2D projections of the 4D Pareto frontier for each algorithm for a fixed random seed on both datasets. We see our proposed algorithm (`qLogNEHVI` BayesOpt) obtains a superior Pareto front, with solutions concentrated towards high safety, high alignment, low cost, and low latency.

MedicalQA. We create our medical dataset using the existing FACTS benchmark (Jacovi et al., 2025) which includes (question, context, answer) triplets.[8] We take the medical Q&A subset, and the authors manually filter out unrepresentative questions. We then combine all the individual context chunks from each question in the medical Q&A split into one large document. The final dataset includes one large medical document, and independent sets of 43 train and 43 test questions.

5.2 Baselines

For our tasks, we lack access to a "ground truth" set of Pareto optimal configurations because we cannot evaluate the objective functions directly, and grid search is computationally unfeasible. Hence we use three baseline approaches that are applicable in MOO settings. The first is *uniform sampling*, which generates configurations independently, where each configuration from the solution space is equally probable. The second is *Sobol sampling*. Like Latin hypercube sampling (Loh, 1996), this method generates configurations that guarantee good high-dimensional uniformity. It is commonly used to initialize optimization algorithms, including BO, and represents a sensible alternative to grid-search. Finally, we use `qLogEHVI` BO, the noiseless variant of our chosen acquisition function (Daulton et al., 2020).

6 Results

We report the HV improvement on the train and test splits of both datasets, across five random seeds in Figure 2. We find that BO methods significantly outperform other baseline approaches on both

[8]FACTS dataset available at: `https://www.kaggle.com/facts-leaderboard`

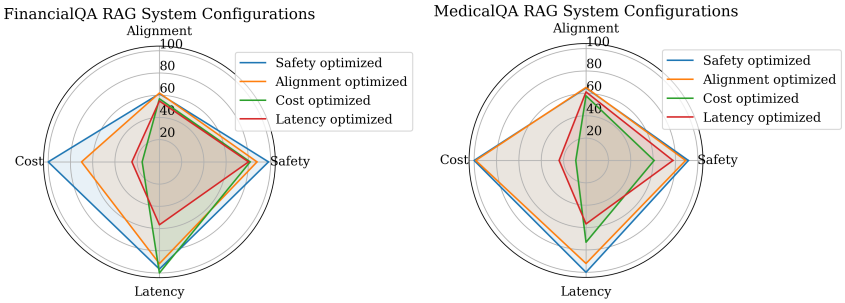

Figure 4: Radar charts comparing the four objective function evaluations for iterations chosen to optimize each objective. We see that improved safety can be achieved at the expense of increased cost and latency. **N.B. Lower is better for cost and latency but higher is better for safety and alignment.**

tasks, and that `qLogNEHVI` outperforms its noiseless variant. Both BO methods show significant improvement compared to baselines after iteration 20, when the BO acquisition function is used to select inputs rather than Sobol sampling.

Figure 3 shows the Pareto front for a fixed seed for the FinancialQA and MedicalQA datasets. Since the overall Pareto frontier lies in $\mathbb{R}^4$ (as there are 4 objectives), we project onto three $\mathbb{R}^2$ plots for visualization purposes. We find that `qLogNEHVI` BO obtains a superior Pareto front compared to the baselines, also finding a wider spread of solutions. In particular, we notice significant clustering around sub-optimal solutions when using `qLogEHVI` BO, as observed by Daulton et al. (2021).

Figure 4 depicts the objective function evaluations across different configurations optimized for each objective. We see that safety and alignment optimized configurations come at the expense of cost and latency, and vice versa. We find that significant cost and latency reduction can be achieved at the cost of minimal safety and alignment reduction. All configuration settings and objective evaluations are detailed in Table 2, where we observe similar chosen parameters across both datasets, especially the choice of LLM and embedding model. From these examples and others, we observe general patterns that help to optimize each objective. Safety optimized configurations often use the large embedding model and a large chunk size. In contrast, latency optimized configurations use the small embedding model and a small chunk size. This is intuitive: high safety requires sufficient high-quality context tokens, whereas low latency necessitates fewer context tokens.

7 Discussion

We frame our discussion as takeaways for industry practitioners that aim to optimize configurations of a RAG pipeline in a multi-objective setting. The first consideration is what we call *objective relationships*. In our experiments, we found that safety and alignment are often positively correlated with each other, and similarly for cost and latency. However, these two sets of objectives involve conflicting parameters, which makes it challenging to set a suitable trade-off between reliability (safety and alignment) and efficiency (cost and latency). Resolving conflicts between objectives is inherently challenging and remains an open question; we recommend that practitioners be thoughtful about latent relationships between objectives when choosing which objectives to optimize over.

The second consideration is *task dependence*, meaning that an optimal configuration for a RAG pipeline on a task in one setting may not generalize to another. While we observe configurations that work well across both tasks with respect to certain objectives (*e.g.,* high chunk size correlates with higher safety and alignment), there is no optimal configuration that is shared across the two tasks. Practitioners should be aware that the optimal configuration will be highly dependent on the task they are building for, including the way their system will be used, the domain, and the stakeholders.

The third consideration is *objective dependence*, where objective evaluations follow different trends (or have no trend) across different configurations. For example, we see high objective dependence

for temperature, since high temperatures (*e.g.,* > 1.0) consistently reduced all four objectives. However, it is harder to discern a relationship between chunk overlap and the objectives, indicating low objective dependence. Task and objective dependence can also compound, highlighting the challenge of collectively optimizing the configuration of a RAG system.

Future Work Our results demonstrate that end-to-end optimization of RAG systems is a promising avenue for research. We highlight two key areas for future exploration: first, improving the efficiency of Algorithm 1. A potential direction is "decoupled evaluation", where not all objectives are assessed at every iteration, towards a cost-aware optimization strategy. For instance, evaluating safety and alignment is significantly more expensive than measuring cost and latency.

Second, improvements can be made to the framework itself. Our current approach does not account for prompt engineering, despite its significant impact on response quality. Additionally, our safety and alignment metrics considered in this work remain limited in scope, excluding aspects such as toxicity and data leakage. However, our framework is inherently flexible and can be extended with adapted objectives or additional parameters. Another open challenge is configuration selection from the Pareto frontier. As Figure 3 illustrates, multiple configurations are Pareto-optimal, making the selection of the most suitable trade-off for a given application non-trivial.

ACKNOWLEDGMENTS

The authors thank Abhinav Raghunathan, Paul Dongha and David More for their guidance and industry expertise. We also thank Eirini Ioannou for her advice and suggestions. This work was supported in part by NSF awards 1916505, 1922658, 2312930, 2326193, and by the NSF GRFP (DGE-2234660). This work was supported in part by ELSA: European Lighthouse on Secure and Safe AI project (grant agreement No. 101070617 under UK guarantee). Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of any of these funding agencies, European Union or European Commission.

REFERENCES

Sebastian Ament, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for bayesian optimization. *Advances in Neural Information Processing Systems*, 36:20577–20612, 2023.

Charles Audet and John E Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on optimization*, 17(1):188–217, 2006.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Eytan Bakshy, Lili Dworkin, Brian Karrer, Konstantin Kashin, Benjamin Letham, Ashwin Murthy, and Shaun Singh. Ae: A domain-agnostic platform for adaptive experimentation. In *Conference on neural information processing systems*, pp. 1–8, 2018.

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The journal of machine learning research*, 13(1):281–305, 2012.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian,

Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. *Advances in Neural Information Processing Systems*, 33:9851–9864, 2020.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement. *Advances in Neural Information Processing Systems*, 34:2187–2200, 2021.

Kalyanmoy Deb, Karthik Sindhya, and Jussi Hakanen. Multi-objective optimization. In *Decision sciences*, pp. 161–200. CRC Press, 2016.

Mike Diessner, Joseph O'Connor, Andrew Wynn, Sylvain Laizet, Yu Guan, Kevin Wilson, and Richard D Whalley. Investigating bayesian optimization for expensive-to-evaluate black box functions: Application in fluid dynamics. *Frontiers in Applied Mathematics and Statistics*, 8: 1076296, 2022.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023.

Jia Fu, Xiaoting Qin, Fangkai Yang, Lu Wang, Jue Zhang, Qingwei Lin, Yubo Chen, Dongmei Zhang, Saravan Rajmohan, and Qi Zhang. Autorag-hp: Automatic online hyper-parameter tuning for retrieval-augmented generation. *arXiv preprint arXiv:2406.19251*, 2024.

Robert B Gramacy. *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. Chapman and Hall/CRC, 2020.

Andreia P Guerreiro, Carlos M Fonseca, and Luís Paquete. The hypervolume indicator: Computational problems and algorithms. *ACM Computing Surveys (CSUR)*, 54(6):1–42, 2021.

Aman Gupta, Anup Shirgaonkar, Angels de Luis Balaguer, Bruno Silva, Daniel Holstein, Dawei Li, Jennifer Marsman, Leonardo O Nunes, Mahsa Rouzbahman, Morris Sharp, et al. Rag vs fine-tuning: Pipelines, tradeoffs, and a case study on agriculture. *arXiv preprint arXiv:2401.08406*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration (extended version). *Technical Report TR-2010–10, University of British Columbia, Computer Science, Tech. Rep.*, 2010.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Alon Jacovi, Andrew Wang, Chris Alberti, Connie Tao, Jon Lipovetz, Kate Olszewska, Lukas Haas, Michelle Liu, Nate Keating, Adam Bloniarz, et al. The facts grounding leaderboard: Benchmarking llms' ability to ground responses to long-form input. *arXiv preprint arXiv:2501.03200*, 2025.

Dongkyu Kim, Byoungwook Kim, Donggeon Han, and Matouš Eibich. Autorag: automated framework for optimization of retrieval augmented generation pipeline. *arXiv preprint arXiv:2410.20878*, 2024.

Bingdong Li, Zixiang Di, Yanting Yang, Hong Qian, Peng Yang, Hao Hao, Ke Tang, and Aimin Zhou. It's morphing time: Unleashing the potential of multiple llms via multi-objective optimization. *arXiv preprint arXiv:2407.00487*, 2024.

Kaiwen Li, Tao Zhang, and Rui Wang. Deep reinforcement learning for multiobjective optimization. *IEEE transactions on cybernetics*, 51(6):3103–3114, 2020.

Wei-Liem Loh. On latin hypercube sampling. *The annals of statistics*, 24(5):2058–2080, 1996.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. On llms-driven synthetic data generation, curation, and evaluation: A survey. *arXiv preprint arXiv:2406.15126*, 2024.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.

Subhojyoti Mukherjee, Anusha Lalitha, Sailik Sengupta, Aniket Deshmukh, and Branislav Kveton. Multi-objective alignment of large language models through hypervolume maximization. *arXiv preprint arXiv:2412.05469*, 2024.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*, 2018.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Christophe Tribes, Sacha Benarroch-Lelong, Peng Lu, and Ivan Kobyzev. Hyperparameter optimization for large language model instruction-tuning. *arXiv preprint arXiv:2312.00949*, 2023.

Chi Wang, Qingyun Wu, Silu Huang, and Amin Saied. Economic hyperparameter optimization with blended search strategy. In *International Conference on Learning Representations*, 2021.

Chi Wang, Xueqing Liu, and Ahmed Hassan Awadallah. Cost-effective hyperparameter optimization for large language model generation inference. In *International Conference on Automated Machine Learning*, pp. 21–1. PMLR, 2023.

Siyin Wang, Shimin Li, Tianxiang Sun, Jinlan Fu, Qinyuan Cheng, Jiasheng Ye, Junjie Ye, Xipeng Qiu, and Xuanjing Huang. Llm can achieve self-regulation via hyperparameter aware generation. *arXiv preprint arXiv:2402.11251*, 2024.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. β-dpo: Direct preference optimization with dynamic β. *arXiv preprint arXiv:2407.08639*, 2024.

Qingyun Wu, Chi Wang, and Silu Huang. Frugal optimization for cost-related hyperparameters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10347–10354, 2021.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhanhui Zhou, Jie Liu, Jing Shao, Xiangyu Yue, Chao Yang, Wanli Ouyang, and Yu Qiao. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 10586–10613, 2024.

A Related Work on Fine-tuning and Model Merging

While we do not explore fine-tuning or model-merging in our work, several authors have studied the effectiveness of HO to improve the alignment process.

Fine-tuning. Wu et al. (2024) explored the importance of hyperparameter tuning the β-parameter[9] for Direct Preference Optimization (DPO) (Rafailov et al., 2024), a popular framework for human-preference tuning LLMs. Significantly, they uncovered settings in which increasing β improves DPO performance, and others where increasing β has the exact opposite effect and decreases performance. Wang et al. (2024) introduce an approach called Hyperparameter Aware Generation (HAG), that allows LLMs to "self-regulate" hyperparameters like temperature, top-p, top-k, and repetition penalty during inference. They observed that different configurations of these hyperparameters lead to different performances on tasks like reasoning, creativity, translation, and math. Tribes et al. (2023) used hyperparameter optimization (HO) to improve the instruction fine-tuning process, adjusting the hyperparameters rank and scaling α for Low-Rank Adaptation (LoRA) (Hu et al., 2021)[10], as well as the model dropout rate and learning rate. In their experiments, they fine-tuned a Llama 2 7B parameter model[11], and found that HO-fine-tuning resulted in better performance on tasks like MMLU (Hendrycks et al., 2020), BBH (Suzgun et al., 2022), DROP (Dua et al., 2019), and HumanEval (Chen et al., 2021), as compared to vanilla fine-tuning. Methodologically, they tested two HO approaches: Tree-structured Parzen Estimator tuning[12] (Bergstra et al., 2011) and Mesh Adaptive Direct Search (Audet & Dennis Jr, 2006), and found better performance with the latter. Overall, their results confirm the necessity of careful HO in instruction-tuning.

Model merging. Li et al. (2024) frame LLM model merging, or combining different "source" (or base) models to create a unified model that retains the strengths of each model, as a multi-objective optimization (MOO) problem. They use parallel multi-objective Bayesian optimization (qEHVI) (Daulton et al., 2020) to search over a range of model merging techniques like Model Soup (Wortsman et al., 2022) and TIES-Merging (Yadav et al., 2024) (and the associated hyperparameters of those techniques), and evaluate the performance of the merged model on benchmarks

[9] β governs the extent to which the policy model's behavior can diverge from the original model.

[10] Low-Rank Adaptation (LoRA) is a method that reduces the number of trainable parameters for a fixed model for downstream tasks like fine-tuning.

[11] `https://huggingface.co/meta-llama/Llama-2-7b`

[12] TPE is a Bayesian optimization (Bergstra et al., 2011) algorithm that uses a probabilistic model for HO. It is a Sequential Model-Based Optimization (SMBO) method (Hutter et al., 2010).

like MMLU and Big-Bench Hard (Suzgun et al., 2022). Our work is distinct from approaches using model merging in that we search over choices for LLMs rather than combine them. However, similar to Li et al. (2024), we test the effectiveness of the qEHVI and qNEHVI (Daulton et al., 2021) (a variation allowing for noisy objectives) for HO.

B Approximating Objective Functions

Equation 3 defines our proposed task as the maximization of the $\mathcal{HV}$ across multiple objectives. Each objective function evaluates a property of the RAG system for a given workload. We define a workload as a probability distribution across all possible queries, $P(q)$, where $q \in \mathcal{Q}$ is a user query for the RAG system. Further, we use $f_m^q : \mathcal{X}, \mathcal{Q} \mapsto \mathbb{R}$ to denote an evaluation function for an individual query and objective m. Using these definitions, we can write down the objective function as the expectation across queries:

$$f_m(\mathbf{x}) = \sum_{q \in \mathcal{Q}} f_m^q(\mathbf{x}, q) P(q) \tag{6}$$

Assuming we can sample queries from the workload, $q \sim P(q)$, we can use a Monte Carlo approximation for each objective function using a sample set $\mathcal{Q}' \subset \mathcal{Q}$:

$$f_m(\mathbf{x}) \approx \frac{1}{|\mathcal{Q}'|} \sum_{q \in \mathcal{Q}'} f_m^q(\mathbf{x}, q) \tag{7}$$

We assume that generating data synthetically using an LLM (Section 4.1) is equivalent to sampling queries $q \sim P(q)$ to obtain $\mathcal{Q}'$. Our algorithm uses Equation 7 as a tractable approximation of the objective functions in Equation 1.

C Experimental details

We use the following search space for hyperparameters:

- $c_s \in \mathbb{Z}^+$: Maximum number of tokens in each document chunk.
- $c_n \in \mathbb{Z}^+$: Number of chunks retrieved from the vector database for each query.
- $o \in \mathbb{Z}^+$: Number of tokens which overlap between adjacent chunks in a document.
- $t \in [0, 1.2]$: Temperature of the LLM when generating responses.
- $r \in [0, 1]$: Rerank threshold used to set the minimum similarity between the context chunk and query, as evaluated by the reranker[13]. Retrieved documents which are below this threshold are ignored and not passed to the LLM as context. If no chunks exceed this threshold, we choose only the highest scoring chunk as context.
- $\ell \in \{\text{gpt-4o}, \text{gpt-4o-mini}, \text{llama-3.2-3B}, \text{llama-3.1-8B}\}$: Choice of LLM used to generate the response.
- $e \in \{\text{text-embedding-3-large}, \text{text-embedding-3-small}\}$: Choice of embedding model when embedding the queries and document chunks.

[13]We use a fixed rerank model `Salesforce/Llama-Rank-V1` provided by TogetherAI for all RAG systems.

D LLM PROMPTS

We use the following prompt with GPT-4o to generate synthetic questions from the financial source document:

```
You are an expert synthetic data generation agent. Look at this PDF
    document containing information on accounting from a leading global
    financial services organization. Generate 50 questions which
    accountants might ask based only on the information provided in the
    document. Example questions are:

Are there any specific disclosures required when transferring HTM
    securities to AFS?
Are there any tax implications to consider when transferring securities
    between categories?
What are the steps to recording a transfer of AFS securities to HTM?
How should Federal agencies implement the new Land standard?
I have a client in bankruptcy, what is the presentation of financial
    statements once they emerge from bankruptcy
How do you determine if a limited partnership is a VIE
can a debt being refinanced with a different lender result in
    modification accounting?
Are there specific criteria for capitalizing costs related to PP&E
    additions?
how do you account for PP&E additions
which examples in the debt and equity handbook illustrate the accounting
    for preferred stock?
Are there any examples of exit fees in investment company accounting?
```

For prompting the LLM during RAG, we use the following prompt from `https://smith.langchain.com/hub/rlm/rag-prompt`:

```
You are an assistant for question-answering tasks. Use the following
    pieces of retrieved context to answer the question. If you don't know
     the answer, just say that you don't know. Use three sentences
    maximum and keep the answer concise.
Question: {question}
Context: {context}
Answer:
```

E ADDITIONAL RESULTS

		FinancialQA Optimized Configurations			
		Safety	**Alignment**	**Cost**	**Latency**
Hyperparam	Embedding model	text-embedding-3-large	text-embedding-3-large	text-embedding-3-large	text-embedding-3-small
	LLM	gpt-4o-mini	Llama-3.1-8B	Llama-3.2-3B	Llama-3.1-8B
	Chunk size	1024	1024	512	1024
	Chunk overlap	512	128	64	64
	Num chunks	3	4	3	3
	Rerank threshold	0.00	0.64	0.24	1.00
	Temperature	0.03	0.12	0.79	0.00
Objective	Safety	**98.1**	87.8	81.6	80.1
	Alignment	61.3	**62.0**	56.8	54.8
	Cost	585	410	**90.8**	145
	Latency	12.4	11.7	12.8	**7.26**

		MedicalQA Optimized Configurations			
		Safety	**Alignment**	**Cost**	**Latency**
Hyperparam	Embedding model	text-embedding-3-large	text-embedding-3-small	text-embedding-3-large	text-embedding-3-small
	LLM	gpt-4o-mini	gpt-4o-mini	Llama-3.1-8B	Llama-3.1-8B
	Chunk size	1024	1024	256	1024
	Chunk overlap	256	512	32	32
	Num chunks	6	6	2	2
	Rerank threshold	0.00	0.22	0.57	0.36
	Temperature	0.00	0.10	0.57	0.00
Objective	Safety	**91.5**	89.1	60.9	77.7
	Alignment	64.8	**65.3**	57.8	61.3
	Cost	1010	997	**92.6**	244
	Latency	17.0	15.6	12.4	**9.62**

Table 2: Input parameters and objective evaluations for individual configurations optimized for each objective. We observe a similar choice of parameters between both datasets, especially the choice of LLM and embedding model.